An Exploration of Axiomatic Approaches to Information Retrieval

Hui Fang
Department of Computer Science
University of Illinois at Urbana-Champaign

ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign

ABSTRACT

Existing retrieval models generally do not offer any guarantee for optimal retrieval performance. Indeed, it is even difficult, if not impossible, to predict a model's empirical performance *analytically.* This limitation is at least partly caused by the way existing retrieval models are developed where relevance is only coarsely modeled at the level of documents and queries as opposed to a finer granularity level of terms. In this paper, we present a new axiomatic approach to developing retrieval models based on direct modeling of relevance with formalized retrieval constraints defined at the level of terms. The basic idea of this axiomatic approach is to search in a space of candidate retrieval functions for one that can satisfy a set of reasonable retrieval constraints. To constrain the search space, we propose to define a retrieval function inductively and decompose a retrieval function into three component functions. Inspired by the analysis of the existing retrieval functions with the inductive definition, we derive several new retrieval functions using the axiomatic retrieval framework. Experiment results show that the derived new retrieval functions are more robust and less sensitive to parameter settings than the existing retrieval functions with comparable optimal performance.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Retrieval models

General Terms: Experimentation

Keywords: Axiomatic model, retrieval heuristics, constraints, formal models, TF-IDF weighting

1. INTRODUCTION

It has always been a significant challenge to develop principled retrieval methods that are effective, robust, and efficient. Although many information retrieval models have been studied [16, 15, 13, 10, 21, 20, 3, 9, 8], they generally do not offer any guarantee for optimal retrieval performance. Non-optimal parameter setting easily causes a model to perform poorly. As a result, heavy parameter tuning is almost always needed to achieve optimal performance on a particular data set.

In a way, this limitation is caused by the way existing retrieval models are developed. Most existing models have been developed based on a "coarse" or "black box" approximation of the notion of relevance at the level of documents and queries. Such approximation conveniently allows us to avoid addressing relevance *directly* at a finer granularity level of terms. For example, in the vector space model, the notion of relevance is assumed to be captured through a similarity measure on a query vector and a document vector, which allows us to conveniently convert the retrieval problem to one mainly involving vector space operations [15]. Similarly, in probabilistic retrieval models, including the language modeling approaches, the notion of relevance is assumed to be captured through a binary random relevance variable and a probabilistic model is defined to associate this variable with some (probabilistic) representation of documents and queries, which again allows us to avoid directly addressing the notion of relevance and conveniently convert the retrieval problem to one involving defining and estimating probabilistic models [8]. The lack of a detailed modeling of relevance makes it difficult for such a model to achieve optimal retrieval performance. Thus, heuristic modification of a retrieval formula and heuristic introduction of additional parameters are often made to improve retrieval performance. To avoid such heuristic modifications, we will need to capture relevance more *directly* and at a finer granularity level of terms.

Our previous work [2] sheds some light on how to model relevance more directly. This work shows that intuitive retrieval heuristics can be formally defined as constraints on retrieval functions and the empirical performance of a retrieval function is tightly related to how well it satisfies these constraints. It is also shown that none of the analyzed retrieval formula can satisfy all the proposed constraints unconditionally. A very interesting question is thus whether we can systematically search for a retrieval function that can satisfy all the desirable constraints and develop new retrieval models in this way.

In this paper, we present a new axiomatic approach to developing retrieval models based on direct modeling of relevance with formalized retrieval constraints. The basic idea of this axiomatic approach is to search in a space of candidate retrieval functions for one that can satisfy a set of reasonable retrieval constraints. There are some previous studies along this direction, mostly based on logic [1, 6, 22], but, as far as we know, none of these studies has resulted in any effective retrieval function. Although the general idea is similar, our approach is completely different from these previous studies both in the space of retrieval functions considered and in the way we specify the constraints (axioms).

One challenge in developing operational retrieval models using such an axiomatic approach is how to appropriately define the search space for retrieval formulas. To constrain the search space, we assume a "bag-of-terms" representation of queries and documents and propose to define a retrieval function *inductively*. Based on such a definition, a retrieval function can be decomposed into three components, referred to as *Primitive weighting function*, *Query growth function* and *Document growth function*, respectively. Thus searching for a good retrieval function boils down to searching for a good formula for each of these three functions in our constrained search space.

The inductive definition scheme provides a common basis to analytically compare different retrieval functions. We compare and analyze three representative existing retrieval functions in this way and find that they share some commonalities in their primitive weighting functions and query growth functions, but they generally differ in the document growth function. The analysis provides an interpretation of the three component functions of the inductive definition scheme. We further generalize these specific component functions to derive new retrieval formulas within the axiomatic framework. We use the intuitive retrieval constraints proposed in [2] and the technique of exploratory data analysis [4, 5] to constrain the choices for the three component functions and derive several new retrieval functions. We implement and test these new functions with a number of representative test sets. The experiment results show that the derived new functions are more robust and less sensitive to parameter settings than the existing retrieval functions with comparable optimal performance.

The rest of the paper is organized as follows. We first present the axiomatic framework in Section 2. In Section 3, we derive new retrieval functions based on our axiomatic framework. We report experiment results for these new functions in Section 4 and conclude in Section 5.

2. AN AXIOMATIC FRAMEWORK

To define an axiomatic framework for information retrieval, we need to define (1) a *search space* of possible retrieval functions; and (2) a set of *retrieval constraints* that any reasonable retrieval function should satisfy. The assumption is that if a retrieval function satisfies all our constraints, the function would likely be effective empirically. The search space must be large enough to include effective retrieval functions, yet small enough for search. So there is clearly a tradeoff. For the constraints, ideally, we want to have as many constraints as possible so that we can effectively prune the search space and find an effective function more easily. In reality, however, as we add more and more constraints, we may introduce bias and some constraints may be too strong or even contradictory. So there is also a tradeoff. We now discuss how we make these tradeoffs.

2.1 Function space

Since a retrieval function is defined on a document and a query, we first need to define our documents and queries. Following the current retrieval models, we assume that both documents and queries are "bags of terms". To make our framework as general as possible, we include all the scoring functions defined on a bag-of-terms representation of documents and queries in our function space.

Formally, let T be the set of all terms. Let query $Q = \{q_1, ..., q_n\}$ and document $D = \{d_1, ..., d_m\}$ be two bags of terms, where $q_i, d_i \in T$, and it is possible that $q_i = q_j$ and $d_i = d_j$ even if $i \neq j$. Our goal is to define a scoring function $S(Q, D) \in \Re$. To help us search through this function space efficiently and define meaningful constraints on the retrieval functions, we propose to define a retrieval function inductively.

We start with the base case, when both the document and query contain only one term.

Base Case: Assume $Q = \{q\}$ and $D = \{d\}$.

$$S(Q, D) = f(q, d) = \begin{cases} weight(q) = weight(d) & q = d \\ penalty(q, d) & q \neq d \end{cases}$$

Function f gives the score of a one-term document and a one-term query and will be referred to as the *Primitive weighting function*. It rewards the document with a score of $weight(q)$ when d matches q and gives it a penalty score of $penalty(q, d)$ otherwise. We will reasonably assume that $\forall t \in T$, $weight(t) > 0$ and $\forall q, \forall d \neq q, penalty(q, d) < weight(q)$.

In the inductive step, we consider the case when a document or a query contains more than one term.

Inductive Step: $\forall Q, D$ such that $|Q| \geq 1$ and $|D| \geq 1$,
(1) Assume $Q' = Q \cup \{q\}$.

$$S(Q', D) = S(Q \cup \{q\}, D) = g(S(Q, D), S(\{q\}, D), q, Q, D)$$

(2) Assume $D' = D \cup \{d\}$.

$$S(Q, D') = S(Q, D \cup \{d\}) = h(S(Q, D), S(Q, \{d\}), d, Q, D)$$

Function g describes the score change when we add a term to a query, and is called the *Query growth function*. When a new term q is added to a query Q, the score of any document for the new query (i.e. $S(Q \cup \{q\}, D)$) would be mainly determined by the score of the document for the old query (i.e. $S(Q, D)$), the score of the document for the added query term (i.e. $S(\{q\}, D)$), and any possible score adjustment determined by D, Q and q. Similarly, function h describes the score change when we add a term to a document, and is called the *Document growth function*.

Unfortunately, without appropriate constraints on the component functions f, g, and h, the inductive definition above does not necessarily define a function since $S(Q, D)$ may be computed in multiple ways depending on how we construct Q and D. Specifically, the value $S(Q, D)$ may be sensitive to the order of adding terms to the query and/or the document. The following theorem gives a set of necessary and sufficient conditions under which $S(Q, D)$ can be guaranteed to be a function.

Theorem 1 *$S(Q, D)$ is a function if and only if all the following conditions holds.*
(1) $\forall Q, D$ and $\forall q, d \in T$,

$$\delta_d(d, D, Q) + \delta_q(q, D \cup \{d\}, Q) = \delta_q(q, D, Q) + \delta_d(d, D, Q \cup \{q\})$$

(2) $\forall Q, D$ and $\forall d_1, d_2 \in T$,

$$\delta_d(d_1, D, Q) + \delta_d(d_2, D \cup \{d_1\}, Q) = \delta_d(d_2, D, Q) + \delta_d(d_1, D \cup \{d_2\}, Q)$$

(3) $\forall Q, D$ and $\forall q_1, q_2 \in T$,

$$\delta_q(q_1, D, Q) + \delta_q(q_2, D, Q \cup \{q_1\}) = \delta_q(q_2, D, Q) + \delta_q(q_1, D, Q \cup \{q_2\})$$

where $\delta_d(d, D, Q) = S(Q, D \cup \{d\}) - S(Q, D)$ is the score change due to the addition of term d to document D, and $\delta q(q, D, Q) = S(Q \cup \{q\}, D) - S(Q, D)$ is the score change due to the addition of a term q to query Q.

Intuitively, these three conditions simply require that $S(Q, D)$ remains the same no matter in which order the terms are added to the query and the document when we compute it. The proof of this theorem involves a straightforward mathematical induction and is omitted due to the space limit.

2.2 Retrieval Constraints

Another important component in the axiomatic framework is the retrieval constraints. In our previous work [2], we proposed six retrieval constraints that any reasonable retrieval formula should satisfy. However, two of them (i.e., LNC2 and TDC) do not appear to be general enough for a general framework. So we only use the other 4 retrieval constraints in our axiomatic framework, which are re-formalized as follows:

Constraint 1: $\forall Q, D$ and $\forall d \in T$, if $d \in Q$, $S(Q, D \cup \{d\}) > S(Q, D)$.
This constraint says that adding one query term to a document must increase the score. It corresponds to the constraints TF-LNC and TFC1 in [2].

Constraint 2: $\forall Q, D$ and $\forall d \in T$, if $d \notin Q$, $S(Q, D \cup \{d\}) < S(Q, D)$.
This constraint ensures that adding a non-query term to a document must decrease the score. It is essentially the LNC1 constraint in [2].

Constraint 3: $\forall Q, D$ and $\forall d \in T$, if $d \in Q$, $\delta_d(d, D, Q) > \delta_d(d, D \cup \{d\}, Q)$.
This constraint says that the amount of increase in the score due to adding a query term d to a document must decrease as we add more and more d's. It is similar to the TFC2 constraint defined in [2].

2.3 Anatomy of Existing Retrieval Functions

In order to obtain some sense about the relationship between the existing retrieval functions and this new way of defining a retrieval function, we rewrite 3 representative existing retrieval functions using the inductive definition schema. The following notations will be used in this section. C_t^D (C_t^Q) is the count of term t in document D (query Q). N is the total number of documents in the collection. $df(t)$ is the number of documents containing term t. $|D|$ is the length of document D. $avdl$ is the average document length in the collection. $p(t|C)$ is the probability of a term t given by the collection language model [23].

2.3.1 Pivoted Normalization (PN)

PN is a representative of effective vector space retrieval functions with the following scoring formula [18, 17]:

$$S(Q, D) = \sum_{t \in Q \cap D} \frac{Piv\_TF(C_t^D)}{Piv\_LN(|D|)} \cdot C_t^Q \cdot ln \frac{N+1}{df(t)},$$

where $Piv\_TF(x) = 1 + ln(1 + ln(x))$, $Piv\_LN(x) = (1-s) + s\frac{x}{avdl}$.

After rewriting, we have

$$\begin{aligned} weight(q) &= ln \frac{N+1}{df(q)} \cdot \frac{1}{Piv\_LN(1)} \\ penalty() &= 0 \\ g() &= S(Q, D) + S(\{q\}, D) \\ h() &= \lambda_1(|D|) \cdot S(Q, D) + \lambda_2(|D|) \cdot \Delta TF(C_d^D) \cdot S(Q, \{d\}) \end{aligned}$$

where $\Delta TF(x) = Piv\_TF(x+1) - Piv\_TF(x)$, $\lambda_1(x) = \frac{Piv\_LN(x)}{Piv\_LN(x+1)}$ and $\lambda_2(x) = \frac{Piv\_LN(1)}{Piv\_LN(x+1)}$.
The decomposition results show that $weight(q)$ is related to an IDF-like discriminative value of q, while $h()$ appears to implement document length normalization and TF normalization.

2.3.2 Okapi

Okapi is an effective retrieval formula representing the classical probabilistic retrieval model [11, 12]:

$$S(Q, D) = \sum_{t \in Q \cap D} ln \frac{N - df(t) + 0.5}{df(t) + 0.5} \times QTF(C_t^Q) \times TF\_LN(C_t^D, |D|),$$

where $QTF(x) = \frac{(k_3+1) \times x}{k_3 + x}$, $TF\_LN(x, y) = \frac{(k_1+1) \times x}{k_1((1-b) + b\frac{y}{avdl}) + x}$, k_1 (between 1.0-2.0), b (usually 0.75), and k_3 (between 0-1000) are constants.

After rewriting, we have

$$\begin{aligned} weight(q) &= ln \frac{N - df(q) + 0.5}{df(q) + 0.5} \cdot TF\_LN(1, 1) \\ penalty() &= 0 \\ g() &= S(Q, D) + \Delta QTF(C_q^Q) \cdot S(\{q\}, D) \end{aligned}$$

$$\begin{aligned} h() &= S(Q, D) + S(Q; \{d\}) \cdot \Delta TF(C_d^D, |D| + 1) \cdot \gamma \\ &\quad + \sum_{t \in D \cap Q} S(Q, \{t\}) \cdot \Delta LN(C_t^D, |D|) \cdot \gamma \\ &= \sum_{t \in D \cap Q - \{d\}} S(Q, \{t\}) \cdot TF\_LN(C_t^D, |D| + 1) \cdot \gamma \\ &\quad + S(Q; \{d\}) \cdot TF\_LN(C_d^D + 1, |D| + 1) \cdot \gamma \end{aligned}$$

where $\Delta TF(x, y) = TF\_LN(x+1, y) - TF\_LN(x, y)$, $\Delta LN(x, y) = TF\_LN(x, y+1) - TF\_LN(x, y)$, $\Delta QTF(x) = QTF(x+1) - QTF(x)$ and $\gamma = \frac{1}{TF\_LN(1,1)}$.
It shows again that $weight(q)$ is an IDF-related value of q. And $h()$ again implements length normalization and TF normalization, though the form of the formula is more complex than in the case of PN.

2.3.3 Dirichlet Prior (DP)

DP is an effective langauge modeling approach [23]:

$$S(Q, D) = \sum_{t \in Q \cap D} C_t^Q \cdot ln(1 + \frac{C_t^D}{\mu \cdot p(t|C)}) + |Q| \cdot ln \frac{\mu}{|D| + \mu}$$

After rewriting, we have

$$\begin{aligned} weight(q) &= ln(1 + \frac{1}{\mu \cdot p(q|C)}) - ln(1 + \frac{1}{\mu}) \\ penalty() &= -ln(1 + \frac{1}{\mu}) \\ g() &= S(Q, D) + S(\{q\}, D) \\ h() &= S(Q, D) + \beta(C_d^D, p(d|C)) \cdot S(Q, d) \\ &\quad + \theta(p(d|C), C_d^D, |D|, |Q|) \end{aligned}$$

where $\theta(x, y, z, l) = l \cdot (ln\frac{z+\mu}{z+1+\mu} - \frac{ln(1+\frac{y+1}{\mu \cdot x}) - ln(1+\frac{y}{\mu \cdot x})}{ln(1+\frac{1}{\mu \cdot x})} \cdot ln\frac{\mu}{1+\mu})$ and $\beta(x, z) = \frac{ln(1+\frac{x+1}{\mu \cdot z}) - ln(1+\frac{x}{\mu \cdot z})}{ln(1+\frac{1}{\mu \cdot z})}$.

The results show that $weight(q)$ is yet again an IDF-related value of q. However, $penalty()$ is not equal to 0 as in the previous two methods; instead, it is a negative value, which also contributes document length normalization. Function h takes yet another complex form, involving not only TF and length normalization but also the IDF-like variable $p(d|C)$. Function θ is playing a role for additional score adjustment due to the addition of the terms.

2.3.4 Summary

The rewriting exercise provides some interesting insights on how we may derive new functions. (1) All the instantiations of $weight(q)$ are related to an IDF-like discrimination value of q. However, $weight(q)$ in Okapi can be smaller than $penalty(q, d)(=0)$, which causes poor performance on verbose queries. (2) There are two ways to implement document length normalization in our framework. The first method is to set $penalty(w, q) < 0$, which would penalize any non-query terms in the document, as in the DP method. The second is to use document length related parameters to adjust the document relevance score as in PN (i.e. $\lambda_1()$ and $\lambda_2()$) and Okapi (i.e. $TF\_LN()$). (3) It shows three possible ways to instantiate the document growth function, which we summarize below in a more general form.

$$\begin{aligned}
S(Q, D \cup \{d\}) &= \lambda_1(|D|) \cdot S(Q, D) + \lambda_2(|D|) \cdot \alpha(C_d^D) \cdot S(Q, \{d\}) \\
S(Q, D \cup \{d\}) &= \sum_{t \in D \cap Q - \{d\}} S(Q, \{t\}) \lambda(|D| + 1, C_t^D) \\
&\quad + S(Q, \{d\}) \cdot \lambda(|D| + 1, C_d^D + 1) \\
S(Q, D \cup \{d\}) &= S(Q, D) + \beta(C_d^D, C_d^Q, p(d|C)) \cdot S(Q, \{d\}) \\
&\quad + \theta(C_d^D, p(d|C), |D|, |Q|)
\end{aligned}$$

3. DERIVATION OF NEW RETRIEVAL FORMULAS

In this section, we study how to instantiate each component function in the framework to derive a new reasonable retrieval function.

3.1 Primitive weighting function

The primitive weighting function has two component functions: $weight(q)$ and $penalty(q, d)$. As discussed in the previous section, the decision on $penalty(q, d)$ affects the instantiation of the document growth function, so we will discuss it later together with the document growth function.

We consider two ways to define $weight(q)$, both connected with how the matching of q contributes to relevance. The first is to define it as the point-wise mutual information between the presence/absence of q in a document ($p(occ)$) and whether the document is relevant to the given query ($p(rel)$).

$$weight(q) = log\frac{p(occ \cap rel)}{p(occ)p(rel)} = log\frac{p(occ|rel)}{p(occ)} \qquad (1)$$

The second is to define it as the conditional probability that a document is relevant if q occurs in the document:

$$weight(q) = P(rel|occ) \qquad (2)$$

$p(occ)$ can be estimated as $p(occ) = \frac{df(q)}{N}$. If the relevance information of documents is available (e.g. through feedback from the users), it would also be easy to estimate $p(occ|rel)$ and $p(rel|occ)$, so $weight(q)$ can be computed accordingly. However, when we have no or insufficient relevance information about documents, it would be hard to compute $weight(q)$ directly. One possible solution is to employ techniques of exploratory data analysis [4, 5]. The basic idea is to find some empirical function that can explain the relationship between such unknown variables and some known variables well on some training data. For example, we may relate $weight(q)$ to the known variables $p(occ)$ and try to find a function of $p(occ)$ that can approximate $weight(q)$ well. Specifically, for a given data set, we compute $weight(q)$ (according to Equation (1) or (2)) and $p(occ)$ for each query term. Since the variance of these variables is large, we follow [4] and group the data points together in bins. We average both known and unknown variables for a bin to obtain a "pseudo data point". Finally, we plot the graph for these two variables (i.e. $weight(q)$ vs. $p(occ)$) for every pseudo data point.

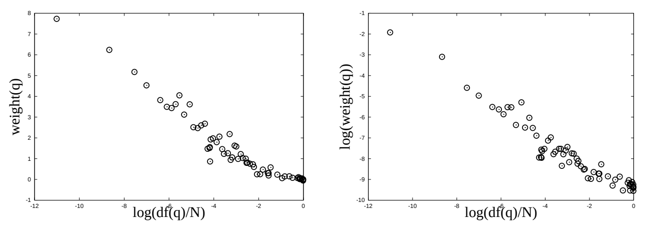

Figure 1: Plot of $weight(q)$(computed using Equation 1) vs. $log\frac{df(q)}{N}$ (Left) and plot of $log(weight(q))$ (computed using Equation 2) vs. $log\frac{df(q)}{N}$ (Right)

In the left plot of Figure 1, we plot the $weight(q)$ computed using Equation(1) against $log(P(occ)) = log\frac{df(q)}{N}$ on some AP data set. (The plots on other data sets are similar.) There appears to be a negative linear correlation between them. Thus we assume $weight(q) = a \log\frac{df(q)}{N} + b$, where a and b are constants. Visually examining several such plots on different data sets indicates $a = -1$ and $b = 0$ may be a good approximation. That is $weight(q) = log\frac{N}{df(q)}$, which will be referred to as *LOG weighting function*. Note that the LOG weighting function is just the typical IDF [19, 14].

The right plot in Figure 1 shows how $\log(weight(q))$, where $weight(q)$ is computed using Equation(2), is related to $log\frac{df(q)}{N}$. We also see a negative linear correlation between them. Again, as a crude approximation, we may assume

$$weight(q) = (\frac{N}{df(q)})^k, (0 < k < 1)$$

where k is a parameter. We call this formula *EXP weighting function*.

3.2 Query Growth Function

The analysis of existing retrieval functions reveals that their query growth functions are quite similar and of a relatively simple form. The slightly more complicated form of Okapi has not shown any clear benefit in our preliminary experiments. Thus we fix our choice of query growth function to the following simple form: $S(Q \cup \{q\}, D) = S(Q, D) + S(\{q\}, D)$.

3.3 Document Growth Function

We generalize the document growth functions of the three existing retrieval functions and explore how to generate some interesting alternative choices.

3.3.1 Formula 1—PN Variation

The generalized form of the PN document growth function is

$$S(Q, D \cup \{d\}) = \lambda_1(|D|) \cdot S(Q, D) + \lambda_2(|D|) \cdot \alpha(C_d^D) \cdot S(Q, \{d\})$$

which is a weighted linear combination of $S(Q, D)$and $S(Q, \{d\})$ with the weights depending on three unknown functions (i.e. $\lambda_1, \lambda_2, \alpha$).

We can easily recover PN with the following instantiations:

$$\begin{aligned}\lambda_1(x) &= \frac{avdl \cdot \frac{1-s}{s} + x}{avdl \cdot \frac{1-s}{s} + x + 1}, \lambda_2(x) = \frac{avdl \cdot \frac{1-s}{s} + 1}{avdl \cdot \frac{1-s}{s} + x + 1} \\ \alpha(y) &= ln(1 + ln(y + 1)) - ln(1 + ln(y)), (y > 1)\end{aligned}$$

We now discuss how we may exploit our inductive definition scheme and retrieval constraints to find some interesting alternative instantiations of λ_1, λ_2 and α.

First, we need to make sure that $S(Q, D)$ is a function. Applying theorem 1, we find that condition (1) and condition (3) can be satisfied unconditionally, but in order to satisfy condition (2), the following two equations must hold.

$$\begin{aligned}\lambda_2(k + 1) &= \lambda_1(k + 1) \times \lambda_2(k), k \geq 0 &(3)\\ \lambda_2(0) &= 1 &(4)\end{aligned}$$

Next, the analysis of constraint 1 suggests that

$$\frac{\lambda_2(k)}{1 - \lambda_1(k)} > \frac{S(Q, D)}{weight(q)}. \quad (5)$$

Since $S(Q, D)$ is roughly a sum of weights over all matched terms, for most documents, we may expect $\frac{S(Q,D)}{weight(q)} < avdl$. Thus we may consider the following somehow stronger, but simpler condition; if Equation (6) holds, we may expect Equation (5) to hold for most documents.

$$\frac{\lambda_2(k)}{1 - \lambda_1(k)} > avdl. \quad (6)$$

Furthermore, Constraint 2 implies that

$$\forall k, \lambda_1(k) < 1. \quad (7)$$

One way to satisfy this condition is to let $\lambda_1(k) = \frac{f(k)}{f(k+1)}$, where $f(k)$ decreases when k increases. A natural simple choice for $f(k)$ is $f(k) = a \times k + b$, where $a > 0$. In this case, $\lambda_1(k) = \frac{a \times k + b}{a \times (k+1) + b}$. According to Equation (3), $\lambda_2(k) = \frac{f(1)}{f(k+1)} = \frac{a+b}{a \times (k+1)+b}, k > 0$. Therefore, Equation (6) is equivalent to $1 + \frac{b}{a} > avdl$. Thus we can assume $\frac{b}{a} = avdl/s$ and $0 < s < 1$. So, we have

$$\lambda_1(k) = \frac{k + \frac{avdl}{s}}{k + 1 + \frac{avdl}{s}}, \lambda_2(k) = \frac{1 + \frac{avdl}{s}}{k + 1 + \frac{avdl}{s}}.$$

Finally, it follows from the analysis of Constraint 3 that $\alpha(C_d^D)$ decreases when C_d^D increases. It is easy to show that $\alpha(0) = 1$. So $\forall x, \alpha(x) \leq 1$. Leaving the study of a better form of $\alpha(C_d^D)$ for our future work, we can simply take the corresponding component from the pivoted normalization formula. That is, $\alpha(k) = ln(1+ln(k+1))-ln(1+ln(k)), k \geq 1$ and $\alpha(0) = 1$.

Using this document growth function together with $penalty(q, d) = 0$, we obtain the following retrieval function

$$S(Q, D) = \sum_{t \in D \cap Q} TF(C_t^D) \cdot C_t^Q \cdot weight(t) \frac{avdl + s}{avdl + |D| \cdot s} \quad (8)$$

where $0 \leq s \leq 1$ and $TF(x) = 1 + ln(1 + ln(x))$.

If we set $s = \frac{s'}{1-s'}$ and $weight(q) = log \frac{N+1}{df(q)}$, Equation 8 turns into PN with parameter s'. The constraint $0 \leq s \leq 1$ is equivalent to $0 \leq s' \leq 0.5$, which is a narrower range than the full range $(0, 1)$ allowed by the standard PN method. Empirical study [2] shows that the optimal value of s' is always smaller than 0.4, thus the new formula we derived using the axiomatic framework has a more reasonable parameter range than the original PN, which is due to the introduction of the extra constraint Equations (5) and (6).

3.3.2 Formula 2—Okapi Variation

The generalized form of the Okapi document growth function is

$$\begin{aligned}S(Q, D \cup \{d\}) &= \sum_{t \in D \cap Q - \{d\}} S(Q, \{t\}) \lambda(|D| + 1, C_t^D) \\ &\quad + S(Q, \{d\}) \cdot \lambda(|D| + 1, C_d^D + 1)\end{aligned}$$

It differs from the document growth function of PN in that the weights of linear combination are related to not only the document length but also the term count and we also have just one unknown function (i.e. λ) to instantiate. The following instantiation clearly recovers Okapi. $\lambda(x, y) = \frac{(k_1+1) \times y}{k_1((1-b)+b\frac{x}{avdl})+y}$.

We now explore how to find any interesting alternative instantiations of $\lambda(x, y)$, where x is related to the document length and y is related to the term count. Again, we check all the constraints to see whether they can provide us more clues about λ.

All the three conditions in Theorem 1 are satisfied unconditionally. Constraint 1 indicates that $\lambda(x + 1, y + 1) > \lambda(x, y)$. The analysis of constraint 2 shows that $\lambda(x+1, y) < \lambda(x, y)$, which means $\lambda(x, y)$ decreases when x increases. From these, it follows that $\lambda(x + 1, y + 1) > \lambda(x + 1, y)$, i.e., $\lambda(x, y)$ increases as y increases. Constraint 3 indicates that $\lambda(x, y)$ should be a sublinear function w.r.t. y. From the Okapi instantiation, it seems that $\lambda(x, y)$ controls how to penalize a long document as well as how to normalize the term frequency for every term. We consider a slightly more general form than the Okapi instantiation, $\lambda(x, y) = \frac{y}{(ax+b)+y}$. The analysis of constraint 1 implies that $\frac{b}{a} > avdl \times r, (0 < r < 1)$ and $0 < b \leq 1$. One way to satisfy this condition is to set $a = s/avdl$ and $b = s$, where $0 < s \leq 1$. Using this document growth function together with $penalty(q, d) = 0$, we obtain

$$S(Q, D) = \sum_{t \in D \cap Q} C_t^Q \cdot weight(t) \cdot \frac{C_t^D}{\frac{s}{avdl} \cdot |D| + s + C_t^D}$$

3.3.3 Formula 3—DP Variation

Different from PN and Okapi, the DP method partially implements length normalization through setting a negative value to $penalty(q, d)$. The generalized form of the DP document growth function is

$$\begin{aligned}penalty(d, q) &< 0 \\ S(Q, D \cup \{d\}) &= S(Q, D) + \beta(C_d^D, C_d^Q, p(d|C)) \cdot S(Q, d) \\ &\quad + \theta(p(d|C), C_d^D, |D|, |Q|)\end{aligned}$$

Setting $penalty() = -ln(1 + \frac{1}{\mu})$ and setting β and θ as follows would recover the DP function.

$$\begin{aligned}\beta(x, y, z) &= \frac{ln(1 + \frac{x+1}{\mu \cdot z}) - ln(1 + \frac{x}{\mu \cdot z})}{ln(1 + \frac{1}{\mu \cdot z})} \\ \theta(x, y, z, l) &= l \cdot (ln \frac{z + \mu}{z + 1 + \mu} - \frac{ln(1 + \frac{y+1}{\mu \cdot x}) - ln(1 + \frac{y}{\mu \cdot x})}{ln(1 + \frac{1}{\mu \cdot x})} \cdot ln \frac{\mu}{1 + \mu})\end{aligned}$$

Table 1: Optimal Performance Comparison of the Derived Formulas

Formula	Trec7				Trec8				Web			
	sk	sv	lk	lv	sk	sv	lk	lv	sk	sv	lk	lv
F1-LOG(Piv)	0.176	0.146	——	0.199	0.245	0.205	——	0.234	0.288	0.212	——	0.214
F1-EXP	0.184	0.173	——	0.211	0.243	0.225	——	0.251	0.288	0.228	——	0.241
F2-LOG	0.185	0.159	——	0.208	0.260	0.210	——	0.240	0.295	0.245	——	0.266
F2-EXP	0.187	0.186	——	0.225	0.257	0.236	——	0.260	0.289	0.272	——	0.292
F3-LOG	0.180	0.154	——	0.204	0.244	0.206	——	0.240	0.290	0.213	——	0.213
F3-EXP	0.187	0.180	——	0.213	0.244	0.227	——	0.250	0.288	0.229	——	0.235

Formula	FR				AP				DOE			
	sk	sv	lk	lv	sk	sv	lk	lv	sk	sv	lk	lv
F1-LOG(Piv)	0.225	0.143	0.269	0.208	0.226	0.193	0.385	0.292	0.179	0.105	0.269	0.210
F1-EXP	0.223	0.144	0.267	0.200	0.223	0.197	0.376	0.278	0.172	0.119	0.269	0.207
F2-LOG	0.223	0.164	0.271	0.241	0.227	0.201	0.386	0.296	0.184	0.110	0.270	0.209
F2-EXP	0.222	0.169	0.268	0.241	0.225	0.203	0.379	0.280	0.175	0.116	0.269	0.203
F3-LOG	0.223	0.141	0.265	0.203	0.227	0.192	0.386	0.295	0.180	0.103	0.266	0.212
F3-EXP	0.218	0.142	0.265	0.191	0.225	0.196	0.377	0.272	0.173	0.111	0.271	0.203

To seek for any interesting alternative instantiations, we follow DP and set $penalty(q,d) = c$ where c is a negative constant. We consider a simple case where $\theta() = 0$ and $\beta()$ is only related to C_d^D and C_d^Q as follows.

$$\beta(C_d^D, C_d^Q, p(d|C)) = \beta'(C_d^D, C_d^Q) \quad = \quad \begin{cases} \alpha(C_d^D), & C_d^Q \neq 0 \\ 1, & C_d^Q = 0 \end{cases}$$

$\beta'(C_d^D, C_d^Q)$ captures the change of term frequency. When d is a query term (i.e. $C_d^Q > 0$) , the change of term frequency is captured by $\alpha(C_d^D)$. As before, the function α can be constrained by Constraint 3. We use the same implementation of $\alpha()$ in PN. On the contrary, when d is a non-query term (i.e. $C_d^Q = 0$), we simply assume that the score change due to the addition of d is always the same. To balance the score between the reward and the penalty, we assume $c = -s/avdl$, where $0 \leq s \leq 1$. We obtain the following hybrid variation of PN and DP:

$$S(Q,D) = \sum_{t \in Q \cap D} C_t^Q \cdot weight(t) \cdot TF(C_t^D) - \gamma(|D|, |Q|)$$

where $TF(x) = 1 + ln(1 + ln(x))$, $\gamma(x,y) = \frac{(x-y)\cdot y \cdot s}{avdl}$ and $0 \leq s \leq 1$

3.4 Derived Retrieval Functions

Combining all the choices, we obtain the following six new retrieval functions.

F1-LOG(s): $S(Q,D) = \sum_{t \in Q \cap D} C_t^Q \cdot TF(C_t^D) \cdot LN(|D|) \cdot LW(t)$

F1-EXP(s,k): $S(Q,D) = \sum_{t \in Q \cap D} C_t^Q \cdot TF(C_t^D) \cdot LN(|D|) \cdot EW(t)$

F2-LOG(s): $S(Q,D) = \sum_{t \in Q \cap D} C_t^Q \cdot TF\_LN(C_t^D, |D|) \cdot LW(t)$

F2-EXP(s,k): $S(Q,D) = \sum_{t \in Q \cap D} C_t^Q \cdot TF\_LN(C_t^D, |D|) \cdot EW(t)$

F3-LOG(s) $S(Q,D) = \sum_{t \in Q \cap D} C_t^Q \cdot TF(C_t^D) \cdot LW(t) - \gamma(|D|, |Q|)$

F3-EXP(s,k) $S(Q,D) = \sum_{t \in Q \cap D} C_t^Q \cdot TF(C_t^D) \cdot EW(t) - \gamma(|D|, |Q|)$

where $TF(x) = 1 + ln(1 + ln(x))$, $LW(t) = ln\frac{N+1}{df(t)}$, $EW(t) = (\frac{N+1}{df(t)})^k$, $LN(x) = \frac{avdl+s}{avdl+x\cdot s}$, $TF\_LN(x,y) = \frac{x}{x+s+\frac{s\cdot y}{avdl}}$ and $\gamma(x,y) = \frac{(x-y)\cdot y \cdot s}{avdl}$, $0 \leq s \leq 1$ and $0 \leq k \leq 1$.

Previous works [14, 24] have also attempted to vary components to form various retrieval formulas in a somehow arbitrary way. Our framework provides more guidances on how to choose the components and can guarantee the performance of the derived functions in some sense.

4. EXPERIMENTS

In this section, we experimentally compare the derived new retrieval functions with the three existing ones. We also examine their parameter sensitivity. Our experiment results show that the new functions can generally achieve comparable optimal performance with the three existing functions, but are more robust and less sensitive to the parameter settings.

4.1 Experiment Design

To cover different types of queries and document sets, we follow [2, 23] and conduct our experiments over six data sets: news articles (AP), technical reports (DOE), government documents (FR), the Web data used in TREC8 (Web), the ad hoc data used in TREC7 (Trec7) and the ad hoc data used in TREC8 (Trec8). For each query, we try different types of queries: short-keyword(SK), short-verbose(SV), long-keyword(LK), and long-verbose (LV). The preprocessing of documents and queries only involves stemming with the Porter's stemmer. We intentionally did not remove stop words for two reasons: (1) A truly robust model should be able to discount the stop words automatically; (2) Removing stop words would introduce at least one extra parameter (e.g. the number of stop words) into our experiments. We set k in the EXP weighting function to 0.35 based on some preliminary experiments.

4.2 Comparison of Derived Functions

To compare the optimal performances of the six derived functions, we vary the parameter value from 0 to 1.0 and select a best run with the highest average precision for each function on each data set. We compare the average precisions of these best runs in Table 1. We make the following observations. First, the optimal performances of all the six functions are comparable, and F1-LOG (i.e., PN) is relatively worse than others. Second, the functions with EXP weighting usually perform better than those with LOG weighting for verbose queries, but worse for keyword queries. Finally, the functions with F2 usually performs better than those with F1 and F3. The parameter sensitivity study also shows that F2-EXP appears to be more stable than others. So, it appears that F2-EXP is overall a better choice than others. Below we compare its performance with existing retrieval functions.

4.3 Comparison with Existing Functions

We compare the performance of one derived function (i.e.F2-EXP) with PN, Okapi, and DP. Due to the poor performance of the original Okapi on verbose queries, we also compare with the modified Okapi (i.e., Okapi with traditional IDF) [2]. Since other retrieval functions all have just one pa-

Table 2: Performance Comparison with Existing Formulas—Top 25-percentile

Formula	**Trec7**				**Trec8**				**Web**			
	sk	sv	lk	lv	sk	sv	lk	lv	sk	sv	lk	lv
F2-EXP	**0.187**	**0.187**	——-	0.224	**0.256**	**0.236**	——-	**0.260**	0.289	**0.272**	——-	**0.291**
F2-EXP-0.5	0.186	0.186	——-	**0.225**	0.250	**0.236**	——-	**0.260**	0.282	**0.272**	——-	**0.291**
Pivoted	0.174	0.145	——-	0.196	0.239	0.201	——-	0.230	0.253	0.207	——-	0.212
Okapi	0.185	0.084	——-	0.073	0.251	0.101	——-	0.108	0.310	0.203	——-	0.229
Mod-Okapi	0.185	0.159	——-	0.215	0.252	0.218	——-	0.253	**0.312**	0.244	——-	0.279
Dirichlet	0.186	0.182	——-	0.224	0.251	0.228	——-	0.259	0.289	**0.272**	——-	**0.291**

Formula	**FR**				**AP**				**DOE**			
	sk	sv	lk	lv	sk	sv	lk	lv	sk	sv	lk	lv
F2-EXP	0.223	**0.167**	0.267	0.234	0.223	0.197	0.377	0.276	0.175	0.114	0.268	0.203
F2-EXP-0.5	0.222	0.164	0.266	0.227	0.220	0.190	0.374	0.272	0.174	0.112	0.268	0.203
Pivoted	0.207	0.139	0.250	0.207	0.225	0.190	0.383	0.288	0.179	0.102	0.263	0.206
Okapi	**0.229**	0.080	**0.276**	0.079	**0.226**	0.082	**0.385**	0.025	**0.184**	0.081	0.265	0.072
Mod-Okapi	0.226	0.162	0.274	**0.251**	**0.226**	0.194	0.384	**0.295**	0.183	0.104	0.270	0.216
Dirichlet	0.206	0.157	0.244	0.233	0.224	**0.204**	0.375	0.292	0.181	**0.125**	**0.276**	**0.228**

Table 3: Performance Comparison with Existing Formulas—Bottom 25-percentile

Formula	**Trec7**				**Trec8**				**Web**			
	sk	sv	lk	lv	sk	sv	lk	lv	sk	sv	lk	lv
F2-EXP	**0.183**	**0.177**	——-	**0.212**	**0.243**	**0.214**	——-	**0.241**	0.275	**0.253**	——-	**0.254**
Pivoted	0.053	0.048	——-	0.077	0.085	0.083	——-	0.095	0.041	0.042	——-	0.051
Okapi	0.161	0.059	——-	0.053	0.223	0.085	——-	0.088	0.215	0.139	——-	0.153
Mod-Okapi	0.165	0.135	——-	0.161	0.227	0.171	——-	0.185	0.223	0.219	——-	0.197
Dirichlet	0.175	0.154	——-	0.202	0.235	0.209	——-	0.240	**0.282**	0.233	——-	0.234

Formula	**FR**				**AP**				**DOE**			
	sk	sv	lk	lv	sk	sv	lk	lv	sk	sv	lk	lv
F2-EXP	**0.216**	**0.150**	**0.258**	**0.205**	0.206	0.160	0.351	0.237	**0.171**	0.100	0.252	0.187
Pivoted	0.061	0.056	0.082	0.070	0.089	0.072	0.184	0.135	0.067	0.039	0.122	0.090
Okapi	0.186	0.054	0.228	0.058	0.208	0.076	0.371	0.023	0.167	0.062	0.250	0.051
Mod-Okapi	0.199	0.132	0.251	0.171	0.211	**0.178**	**0.375**	**0.270**	0.170	0.095	0.257	0.186
Dirichlet	0.189	0.138	0.202	0.171	**0.212**	**0.178**	0.357	0.267	0.159	**0.114**	**0.259**	**0.207**

rameter, we set $k_1 = 1.2$, $k_3 = 1000$ and only vary the value of b in Okapi and modified Okapi. For every method, we randomly sample 12 values within the range of the parameter. Skewed samples with 25% or more of the values falling into an interval of 0.1 are discarded. For each method on each collection, we select the top/bottom 25-percentile runs (i.e., 3 runs with the best/worst average precision) from the 12 runs for comparison.

The results are shown in Table 2 (top 25-percentile) and Table 3 (bottom 25-percentile). F2-EXP0.5 is F2-EXP with a fixed value of 0.5 for s. From Table 2, we see that the optimal performance of F2-EXP is quite comparable with that of all the existing retrieval formulas. Even the performance of the derived formula with a fixed parameter value (i.e. F2-EXP-0.5) is also comparable, demonstrating the robustness of this axiomatic retrieval function. The robustness is further confirmed in Table 3, where we see that F2-EXP mostly outperforms others and in Table 4, where we see that the average variance of all 12 runs for F2-EXP is mostly smaller than for all others. It is interesting to note that modified Okapi always performs better than F2-EXP on AP; indeed, AP and DOE seem to be the only data sets where F2-EXP has not shown advantages. Further analysis and experiments are clearly necessary to better understand this.

4.4 Parameter Sensitivity

We compare the parameter sensitivity between F2-EXP, PN Okapi, and modified Okapi. We did not include DP because its parameter is in a different scale, but it is known that its performance is sensitive to the smoothing parameter [23]. We vary the parameter from 0 to 1. The results on TREC7 are shown in Figure 2. The plot demonstrates the stability of F2-EXP, which we have also observed in the plots for other data sets and query types.

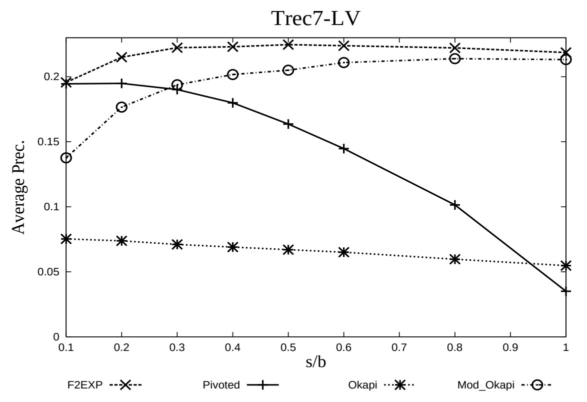

Figure 2: Performance Sensitivity on Trec7-LV

5. CONCLUSIONS AND FUTURE WORK

In this paper, we present a novel axiomatic framework for developing information retrieval models, in which the notion of relevance is directly captured by retrieval constraints. The framework consists of an inductive scheme for function definitions and a set of formalized retrieval constraints. Our work can be regarded as an extension of some previous study [2] to seek for a reasonable retrieval function that can satisfy all the desired retrieval constraints.

The inductive definition scheme provides a common basis to analytically compare different retrieval functions. We

Table 4: Performance Comparison with Existing Formulas—Average variance

Formula	Trec7				Trec8				Web			
	sk	sv	lk	lv	sk	sv	lk	lv	sk	sv	lk	lv
F2-EXP	**5.6e-06**	**2.2e-05**	——-	**3.7e-05**	**3.8e-05**	**1.1e-04**	——-	**6.9e-05**	**3.4e-05**	**7.3e-05**	——-	**2.7e-04**
Pivoted	2.5e-03	1.7e-03	——-	2.4e-03	4.1e-03	2.4e-03	——-	3.1e-03	7.4e-03	4.6e-03	——-	4.4e-03
Mod-Okapi	7.0e-05	1.6e-04	——-	7.3e-04	1.1e-04	4.7e-04	——-	9.8e-04	1.4e-03	1.3e-04	——-	1.4e-03
Dirichlet	1.9e-05	2.8e-04	——-	1.6e-04	4.5e-05	1.5e-04	——-	6.9e-05	1.3e-04	5.2e-04	——-	6.9e-04

Form.	FR				AP				DOE			
	sk	sv	lk	lv	sk	sv	lk	lv	sk	sv	lk	lv
F2-E.	**9.2e-06**	**5.7e-05**	**1.5e-05**	**1.4e-04**	4.6e-05	2.3e-04	1.2e-04	2.6e-04	**4.1e-06**	3.2e-05	5.2e-05	**5.3e-05**
Piv.	3.6e-03	1.2e-03	4.6e-03	3.3e-03	3.2e-03	2.5e-03	7.2e-03	4.2e-03	2.3e-03	7.1e-04	3.7e-03	2.3e-03
M-Ok.	1.4e-04	2.2e-04	8.4e-05	1.3e-03	4.9e-05	4.9e-05	**1.6e-05**	**1.3e-04**	3.3e-05	**1.3e-05**	**2.9e-05**	1.8e-04
Dir.	6.0e-05	9.5e-05	3.8e-04	9.2e-04	**2.7e-05**	**2.2e-04**	5.8e-05	2.4e-04	7.3e-05	3.9e-05	5.3e-05	7.9e-05

compare and analyze three representative existing retrieval functions in our framework and find that while the three functions implement similar heuristics, they implement them in different ways.

We further derive new retrieval functions using the axiomatic framework. We use both intuitive retrieval constraints and exploratory data analysis to guide us in instantiating the three components of the inductive definition and obtain several new retrieval functions. We evaluate these new retrieval functions on a number of representative test sets. The experiment results show that the derived new functions are more stable than the existing retrieval functions with comparable optimal performance.

The axiomatic framework opens up many new possibilities for exploring and developing principled retrieval models based on direct modeling of relevance through constraints. This paper has only moved a very small step in this direction. There are many interesting future research directions. First, we have used only three basic constraints when deriving new retrieval functions. Presumably, with more reasonable constraints, the derived functions will be more specialized and performing better. It would be interesting to add additional constraints, including the two constraints that we have not used from [2]. Second, it would be very interesting to study what constraints are appropriate for modeling relevance/pseudo feedback, which often leads to significant performance improvement over a simple non-feedack retrieval function. Finally, it would be interesting to study theoretical properties of retrieval functions along a similar line to a related work on clustering algorithms [7].

6. ACKNOWLEDGMENTS

This material is based in part upon work supported by the National Science Foundation under award number IIS-0347933. We thank Tao Tao and three anonymous SIGIR reviewers for their useful comments.

7. REFERENCES

[1] P. D. Bruza and T. Huibers. investigating aboutness axioms using information fields. In *Proceedings of the 1994 ACM SIGIR Conference on Research and Development in Information Retrieval*, 1994.

[2] H. Fang, T. Tao, and C. Zhai. A formal study of information retrieval heuristics. In *Proceedings of the 2004 ACM SIGIR Conference on Research and Development in Information Retrieval*, 2004.

[3] N. Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243–255, 1992.

[4] W. R. Grieff. A theory of term weighting based on exploratory data analysis. In *Proceedings of the 1998 ACM SIGIR Conference on Research and Development in Information Retrieval*, 1998.

[5] F. Hartiwig and B. E. Dearing. *Exploratory Data Analysis.* Sage Publications, 1979.

[6] T. Huibers. Towards an axiomatic aboutness theory for information retrieval. *Information Retrieval, Uncertainty and Logics-Advanced Models for the representation and retrieval for information*, 1998.

[7] J. Kleinberg. An impossibility theorem for clustering. In *Advances in NIPS 15*, 2002.

[8] J. Lafferty and C. Zhai. Probabilistic relevance models based on document and query generation. In W. B. Croft and J. Lafferty, editors, *Language Modeling and Information Retrieval.* Kluwer Academic Publishers, 2003.

[9] J. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the ACM SIGIR'98*, pages 275–281, 1998.

[10] S. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146, 1976.

[11] S. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of SIGIR'94*, pages 232–241, 1994.

[12] S. E. Robertson, S. Walker, S. Jones, M. M.Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In D. K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, pages 109–126, 1995.

[13] G. Salton. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer.* Addison-Wesley, 1989.

[14] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24:513–523, 1988.

[15] G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[16] G. Salton, C. S. Yang, and C. T. Yu. A theory of term importance in automatic text analysis. *Journal of the American Society for Information Science*, 26(1):33–44, Jan-Feb 1975.

[17] A. Singhal. Modern information retrieval: A brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 24(4):35–43, 2001.

[18] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *Proceedings of the 1996 ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–29, 1996.

[19] K. Sparck Jones. A statistical interpretation of term specifity and its application in retrieval. *Journal of Documentation*, 28(1):11–22, 1972.

[20] H. Turtle and W. B. Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3):187–222, 1991.

[21] C. J. van Rijbergen. A theoretical basis for theuse of co-occurrence data in information retrieval. *Journal of Documentation*, pages 106–119, 1977.

[22] K.-F. Wong, D. Song, P. Bruza, and C.-H. Cheng. Application of aboutness to functional benchmarking in information retrieval. *ACM Transactions on Information Systems*, 19(4):337–370, 2001.

[23] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of SIGIR'01*, pages 334–342, Sept 2001.

[24] J. Zobel and A. Moffat. Exploring the similarity space. *SIGIR Forum*, 31(1):18–34, 1998.